International Journal of Scientific Engineering and Science
Volume 4, Issue 8, pp. 45-51, 2020. ISSN (Online): 2456-7361

Analysis of Web Application Security Vulnerabilities: A Case Study of Web Applications in Afghanistan

Mohammad Mustafa Naier (MSc)[1], Abdullah Hamidi (MSc)[2], Rafiullah Momand (MSc)[3]
[1]Lecturer at Computer Science Faculty of Polytechnic University, Kabul-Afghanistan
[2]Lecturer at Computer Science Faculty of Herat University, Herat – Afghanistan
[3]Lecturer at Computer Science Faculty of Kabul University, Kabul-Afghanistan

Abstract— *Governments, businesses, and developers are working hard to maintain the security of information on web applications. Though, there are hackers who want to exploit and access classified information from web applications. They use the web application known vulnerabilities to exploit them in order to do what they desire. These vulnerabilities come from the development process of these web applications. This study seeks to answer the question: How to recognize and mitigate most the common vulnerabilities in web applications in the context of Afghanistan cyberspace? Answering this question will help the developers, businesses, and users that which are the top vulnerabilities of web applications, how to prevent exploiting them, what effect it might put on our web application, what characteristics a secure web application has, and what techniques are used to find these vulnerabilities. OWASP introduced a risk assessment framework to calculate the risk. Based on risk assessment framework, 135 websites and web applications are scanned for security issues and vulnerabilities. The result shows that the amount of high risk and medium risk vulnerabilities are seriously more considerable in Afghanistan.*

Keywords— *Web application, Security Vulnerabilities, Afghanistan Cyberspace.*

I. Introduction

Availability of different services are happing through web sites and interactive web applications. Web application carried big amount information in different field of studies, easy process of documents in government, and easy banking payments. Web application enables to share different kind of information; classified and non-classified. For non-classified information limited security can be applied. Non-classified information is general information to the public. Meanwhile for classified information the risk of disruption, destruction and loss can be catastrophic to government and other organization. In both cases, security of information shared through web applications is crucial for government and other organizations either for classified and non-classified information.

Accessibly of information for public is through public network the Internet that carried security challenges. The hackers always trying to exploit web application an access classified information for different purposes. The hackers exploit different vulnerabilities exists in web applications. The OWASP foundation [1] provides top ten up to date web application vulnerabilities in public platform. The developers and hackers can easily access such information which both use for securing and exploiting vulnerabilities.

In the last two decades, different web application for the purpose of information sharing and retrieving information from public is designed in Afghanistan. Different universities in Afghanistan have computer science faculty which they graduate students every year. The graduates are working actively as a developer. According to a study in Afghanistan [2] limited graduates are familiar and implement secure web applications. Therefore, their web applications could be easily hacked by unauthorized people.

In this research different types of vulnerabilities and types of risks are discussed and explained. Based on risks analysis 135 websites and web applications are scanned for security issues and vulnerabilities. The result shows that the amount high risk and medium risk vulnerabilities are seriously considerable. While preforming the exploitation web applications of Afghanistan, different sensitive classified information is easily accessed. In this research such sensitive classified information is not shared but the amount of different high, medium, low and informative vulnerabilities is discussed comprehensively.

II. Web Application Security

Security is the central problem for every web application as all kinds of user access websites and seek to harm the web services. Various kinds of securing techniques are applied to save the websites and web application from attacks or vulnerabilities [3]. To combat security vulnerabilities [4] Functional Programming Way to Communicate with Software Attacks and Vulnerabilities suggested a functional programming path to distinguish and communicate with the software attacks and vulnerabilities.

Web security scanners [5] are used to identify vulnerabilities in web services. “Different scanners have different performance on vulnerabilities detection on various web application”[6].

A. Web Application Vulnerabilities

Vulnerabilities in a system could cause to compromise the security of a complete web applications [7]. Attackers usually trying to find the weakness of web applications through knowledge of application vulnerabilities to benefit from it [7]. The exploitation of such vulnerabilities aids a cybercrime [7]. “According to Gartner Security, the application layer currently contains 90% of all vulnerabilities” [7].

III. Top 10 Web vulnerabilities

Open Web Application Security Project (OWASP)

provides latest information and guidelines about the most important web application vulnerabilities which any organization should follow to limit the flaws for web applications [8]. While developing, purchasing and creating new APIs for any web application, the OWASP security vulnerabilities must be considered [8]. OWASP Top 10 is a strong awareness document for web application security. It represents a common agreement about the most significant security risks two web applications [1].
The OWASP top ten security risks are:

A. Injection

According to OWASP 2017 release [1], Injection happens when an attacker want to insert some codes to access or retrieve data. There are different types of injections, the common ones are SQL, LDAP, XPath. Additionally, NoSQL, XML, OS commands, SMTP parsers and ORM queries are also very important to mention. They are simple to identify when examining the code. Scanners can support attackers locate injection defects. The result of the mentioned injection can be catastrophic for the organization.

The injection can sometimes drive to complete host takeover. The business impression depends on the inadequacies of the purpose and data.

- **SQL injection:** SQL injection is a common way of attacking web applications [9]. Web applications use authentication service to grant users which are stored in database. The attacker uses this opportunity to compromise the website with the username and password. If SQL injection is not considered well, web applications are vulnerable to unauthorized users, therefore obtaining access to use the information of authorized users will have serious effect. A large amount of data in databases are vulnerable to a SQL injection attack which has the strength to completely destroy the data [9].
- **SQL injection types:** SQL Injection attack is a kind of code injection technique which exploits the vulnerability present in the application code for gaining the unauthorized access over the data. The outcomes of SQL injection attacks are the alteration of data, destroying some fields of data, illegal access to data, stealing of data, leaking down the entire database, so on [9].
- **Tautology Queries:** These queries include tautology statements into the queries to gain the Boolean value as always true to get unauthorized access over the database [9]. Example:

  ```
  SELECT * FROM user WHERE uname = 1 or 1=1 AND passwd = abc;
  ```

- **Illegal Queries:** This kind of attack introduces some logically incorrect words into the queries to yield the information about structure of the database [9]. Example:

  ```
  SELECT * FROM user WHERE uname =111 AND password = abc AND CONVERT (int, (SELECT name FROM system WHERE xtype = u));
  ```

- **Union Queries:** This kind of attack attaches a sub query with the existing query using a keyword union in order to perform some malicious activity over the database [9]. Example:

  ```
  SELECT * FROM user WHERE uname = 111 UNION SELECT CardNumber from credit WHERE uname=admin AND passwd = abc;
  ```

- **Piggy bagged Queries**: This kind of attack tries to insert malicious query along with the existing query by adding; at the end of the query through which malicious effects are created [9]. Example:

  ```
  SELECT * FROM user WHERE uname = 111 and passwd = abc; DROP TABLE user;
  ```

- **Stored Procedures:** Queries packed inside the stored procedures are also found vulnerable to piggy bagged queries through which unwanted queries are also allowed to be executed with the existing one [9]. Example:

  ```
  CREATE PROCEDURE DB @uname, @passwd, AS EXEC (SELECT * FROM user WHERE id= +@uname+ and password = +@passwd+);
  ```

B. Broken Authentication

Broken authentication and session management vulnerabilities are general and have a critical in influence once exploited. It enables the attacker to steal privileged user accounts also cover their actions. This vulnerability is created because developers use custom authentication schemes, these schemes contain security flaws in password management, logging out, account update, and timeouts. Detection of broken authentication can be difficult that each customer authentication and session management implementation is unique [10].

C. Sensitive Data Exposure

Overhead the last several years, that has been the most popular impactful attack. The common flaw is easily not encrypting sensitive data. When crypto is applied, weak key generation and administration, and weak algorithm, protocol, and cipher usage are popular, especially for weak password hashing storage techniques. For data in transition, server-side vulnerabilities are essentially easy to detect, but hard for data at rest [1][10].

D. XML External Entities (XXE)

"Several older XML processors allow term of an external entity, a URI that is evaluated during XML processing" [1]. SAST tools can discover this issue by examining dependencies and configuration. DAST tools require extra standard steps to identify and exploit this problem. Manual testers need to be prepared in how to examine for XXE, as it not generally tested as of 2017. These weaknesses can be applied to extract data, execute a remote request from the server, scan internal systems, perform a denial-of-service attack [1].

E. Broken Access Control

In web application there are different level of access control which mostly every developer tries to full fill it. On the other hand, there are web application security testers which can perform automatic discovery to find weaknesses of applications. Additionally, there is manual testing which can detect access control weaknesses. The second way of testing access control in a web application to find weakness is

through manual testing and different methods such as (GET vs PUT), direct object references are included.
[1].

F. Security Misconfiguration

Misconfiguration can be in any level which can compromise the web applications. The levels can be in network services, web server configuration, application server configuration, database and database integration, frameworks used for developing, custom code without considering standards, and pre-installed virtual machines, containers, or different kind of storage. Automatic scanners are helpful for detecting misconfigurations, use of default accounts or configurations, additional assistance, legacy benefits [1][11].

G. Cross-Site Scripting (XSS)

Cross-Site Scripting (XSS) attack is ranked the seventh most significant web application security risk in OWASPs 2019 top 10 [12]. It is very general and can create a moderate influence, such as execute scripts into a victim's browser to hijack user sessions, destroy websites, inject hostile content, and redirect users to malicious sites. XSS vulnerabilities are injections inside the HTML output that the web application generates, which allows the attacker to insert the script into web pages observed by others [10][1][13][14].

H. Insecure Deserialization

It is possible that attackers supply decentralizes hostile or tampered objects which results in vulnerable applications and APIs. The consequence of two main kinds of attacks:

- Attacks related to object and data structure: In this type of attack, the attacker modifies application logic or execute arbitrary remote code if the applications include classes that can change the behavior during or after deserialization.
- Typical data tampering attacks: An example would be access-control-related attacks, where the same data structures are used but the content could be changed.

"Serialization may be used in applications for:

- Remote-and inter-process communication (RPC/IPC)
- Wire protocols, web services, message brokers
- Caching/Persistence
- Databases, cache servers, file systems
- HTTP cookies, HTML form parameters, API authentication tokens". [1][15].

I. Using Components with Known Vulnerabilities

For faster application developments, developers use different built-in and pre-developed component. If the component is out dated or have weakness, it compromises data leakage [15].

J. Insufficient Logging Monitoring

Insufficient logging monitoring allows attackers to attack the systems and compromise or destroy all data [12][16]. Studies of breaches [12] shows the time to detect a breach is over 200 days. Logging monitoring typically detected by external parties rather internal [12].

IV. Methodology of Defining Risks

Vulnerability and risk ranking are very important alongside finding vulnerabilities. Before putting a web application on the production mode to check them and find vulnerabilities we can find some of them using code review penetration testing, you can use also threat modeling for finding the vulnerabilities. Different documents [13][17][18][19] discuss risk analysis framework. In this study the OWASP risk rating methodology is using, the formula for the risk ranking or vulnerability ranking is: Risk = Likelihood * Impact

OWASP risk rating methodology introduce six comprehensive steps to calculate risk:

A. Step 1: Identifying a Risk

The initial identification of security risk is to rate the value of all assets [20]. The tester prerequisites is to gather evidence about the threat agent involved, the attack that will be used, the vulnerability involved, and the influence of a successful exploit on the business [20]. There is the possibility of multiple possible groups of attackers or even multiple possible business impacts [20]. It is essential to find the worst case scenario of the possible impact that will result the highest risk which will destroy the company's asset [20].

B. Step 2: Factors for Estimating Likelihood

While the possible risk in a system is specified, the next step is to find the seriousness of to estimate the "likelihood" [20]. It is the process to check how likely a particular vulnerability is to be discovered and exploited by the attackers [20]. This estimation is not required to be over-precise. It is needed to check whether the likelihood is low, medium, or high [20]. Various factors can help testers to determine the likelihood [20]. "Note that there may be multiple threat agents that can exploit a particular vulnerability, so it's usually best to use the worst-case scenario" [20]. As an example, "an insider is more likely an attacker than an anonymous outsider, but it depends on a number of factors" [20]. "Every factor has a set of possibilities, and each possibility has a likelihood rating from 0 to 9 associated with it" [20]. These numbers will be used later to assess the overall likelihood [20].

- **Threat Agent Factors**
  The initial set of causes are interrelated to the threat agent involved [20]. While finding the mentioned causes, it will help to have the estimation of a successful attack or define the likelihood by this collection of threat agents [20].
- **Skill level**
  How much skills threat agents' group have? [20]. "No technical skills (1), some technical skills (3), advanced computer user (5), network and programming skills (6), security penetration skills (9)" [20].
- **Motive**
  Are the threat agents motivated to find and exploit this vulnerability and how much? "Low or no reward (1), possible reward (4), high reward (9)" [20].
- **Opportunity**
  Which resources and opportunities are required for the threat agents' group to exploit a vulnerability [20]? "Full

access or expensive resources required (0), special access or resources required (4), some access or resources required (7), no access or resources required (9)" [20].

- **Size**
  How many people are involved in the threat agents' group? [20]? Developers (2), system administrators (2), intranet users (4), partners (5), authenticated users (6), anonymous Internet users (9) [20].
- **Vulnerability Factors**
  The following set of factors are associated to the vulnerability involved [20]. The goal is to estimate the likelihood of specific vulnerabilities being discovered and exploited [20].
- **Ease of discovery**
  "How easy is it for this group of threat agents to discover this vulnerability" [20]? "Practically impossible (1), difficult (3), easy (7), automated tools available (9)" [20].
- **Ease of exploit**
  "How easy is it for this group of threat agents to actually exploit this vulnerability" [20]? "Theoretical (1), difficult (3), easy (5), automated tools available (9)" [20].
- **Awareness**
  "How well known is this vulnerability to this group of threat agents" [20]? Unknown (1), hidden (4), obvious (6), public knowledge (9) [20].
- **Intrusion detection**
  "How likely is an exploit to be detected" [20]? "Active detection in application (1), logged and reviewed (3), logged without review (8), not logged (9)" [20].

C. Step 3: Factors for Estimating Impact

There are two kinds of impacts in a successful attack. The first is the "technical impact" on the application, the data it uses, and the functions it provides. The other is the "business impact" on the business and company operating the application [19] [20]. Eventually, the business impact is more important [20]. However, you may not have access to all the information obligatory to figure out the business significances of a successful exploit [20]. In this case, providing as much detail about the technical risk will enable the suitable business characteristic to make a decision about the business risk. Again, each factor has a set of possibilities, and each possibility has an impact rating from 0 to 9 associated with it. We'll use these numbers later to estimate the overall impact [20].

- **Technical Impact Factors**
  "Technical impact can be broken down into factors aligned with the traditional security areas of concern: confidentiality (1), integrity (2), availability (3), and accountability (4)" [19].
- **Loss of confidentiality**
  It is about the amount of data which is revealed and their sensitivity, minimal non-sensitive (2), minimal critical (6), extensive non-sensitive (7) and extensive critical data (9) disclosed [11][17][20].
  - **Loss of integrity**
    How much data is corrupted or damaged? It discusses
    - Minimal lightly corrupted data (1)
    - Minimal seriously corrupted data (3)
    - Extensive slightly corrupted data (5)
    - Extensive seriously corrupted data (7)
    - All data totally corrupted (9).
  - **Loss of availability**
    It discusses about the amount of services interrupted and their importance and includes:
    - "Minimal secondary services interrupted (1)
    - Minimal primary services interrupted (5)
    - Extensive secondary services interrupted (5)
    - Extensive primary services interrupted (7)
    - All services completely lost (9)" [20].
  - **Loss of accountability**
    It is about the traceability of threat agents' actions to individuals. It includes:
    - Fully traceable (1)
    - Possibly traceable (7)
    - Completely anonymous (9) attacks.
- **Business Impact Factors**
  The business influence stems from the technical result but needs a deep understanding of what is important to the company running the application. In general, you should be trying to maintain your risks with business impact, especially if your audience is the administrative level. The business risk is what supports property in fixing security problems. Many companies have an asset allocation guide and/or a business impression reference to help formalize what is important to their business. These criteria can help you focus on what's really necessary for security. The following parts are the most important areas for many businesses which have to be considered [20]:
  - **Financial damage**
    How much financial damage caused by an exploit or attack? "Less than the cost to _x the vulnerability (1), minor effect on annual profit (3), significant effect on annual profit (7), bankruptcy (9)" [20].
  - **Reputation damage**
    How much harm an attack can provide for a business? It includes: "Minimal harm (1), Loss of major accounts (4), loss of goodwill (5), brand damage (9)" [20].
  - **Non-compliance**
    How much exposure does non-compliance introduce? "Minor violation (2), clear violation (5), high profile violation (7)" [20].
  - **Privacy violation**
    How much personally identifiable information could be disclosed? One person (3), about 100 people (5), thousands of people (7), millions of people (9).

D. Step 4: Determining the Severity of the Risk

In this step, the likelihood calculation and the impact estimate are put together to calculate an overall severity for this risk. This is done by figuring out whether the likelihood is low, medium, or high and then do the same for contact. The 0 to 9 scale is split into three parts [20].

E. Step 5: Deciding What to Fix

After the opportunities for the purpose have been assigned there will be a prioritized list of what to x. As a global law, the most severe risks should be fixed first. It really doesn't help the overall risk profile to fix less important risks, even if they're easy or cheap to fix. Remember that not all risks are worth fixing, and some loss is not only expected, but legitimate based on the cost of fixing the issue. For example, if it would cost $ 100,000 to implement controls to stem $ 2,000 of fraud per year, it would take 50 years to return on loan to stamp out the loss. But remember there may be credit damage from the fraud that could cost the organization much more [20].

F. Step 6: Customizing Your Risk Rating Model

Selecting a risk ranking framework which is customizable is critical for a business. A simple model is much more likely to produce results that match people's judgments about what is a dangerous risk. A lot of time can be decreased arguing about the risk ratings if they are not maintained by a model like this. There are several ways to tailor this model for the organization [20].

- **Adding factors**
  Testers could decide which factors are representing what's important for an organization. As an example, impact factors related to loss of human life or some classified information are added to a military application [20].
- **Customizing options**
  There exists some options included in each factor, but the model would be much more effective if the tester can customize these options according to the business. As an example using different names for different classifications and also change the scores associated with the options. [20].
- **Weighting factors**
  According to the above model it is assumed that all the factors are equally important. You can weigh the factors to emphasize the factors that are more significant for the specific business. This will make the model more complicated. But otherwise, everything works the same [20].

V. Afghanistan Government Web-Sites as Target

In this research 135 .AF web sites are selected as a target against top ten OWASP web application security vulnerabilities. All 135 websites are scanned to highlight the vulnerabilities of high, medium and low. The graph highlights the result. The amount of vulnerabilities is considerable, especially the high and Medium.

According to the diagram, about 92% of the tested websites had below 3 High-Risk Vulnerabilities. Only about 8% of them had 4 or more High-Risk Vulnerabilities.

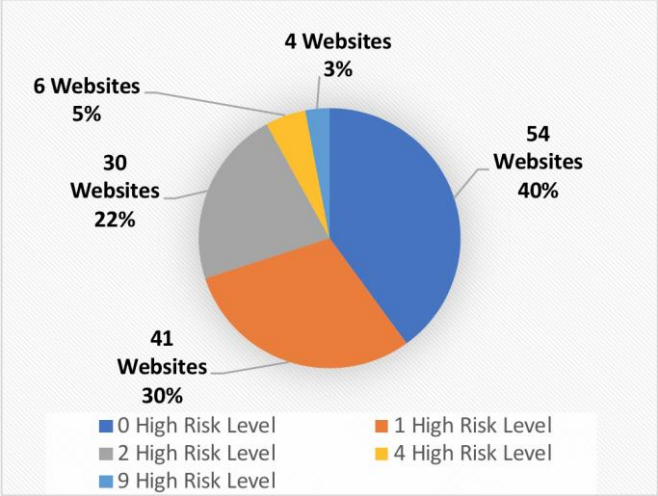

Fig. 1. Details of High-Risk Level for 135 tested websites

Details of Medium Risk level:

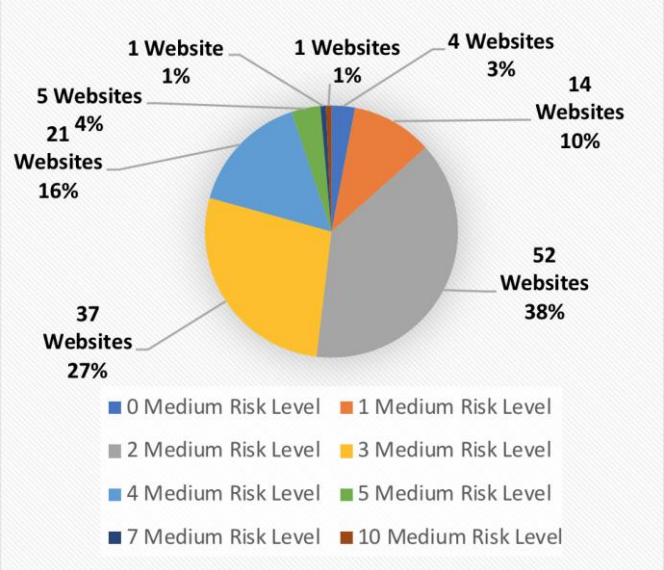

Fig. 2. Details of Medium Risk Level for 135 tested websites

According to the diagram, only 13% of the tested websites had below 3 Medium-Risk Vulnerabilities. In the other hand about 87% of them had 4 - 10 Medium-Risk Vulnerabilities.

Details of Low-Risk level:

International Journal of Scientific Engineering and Science
Volume 4, Issue 8, pp. 45-51, 2020. ISSN (Online): 2456-7361

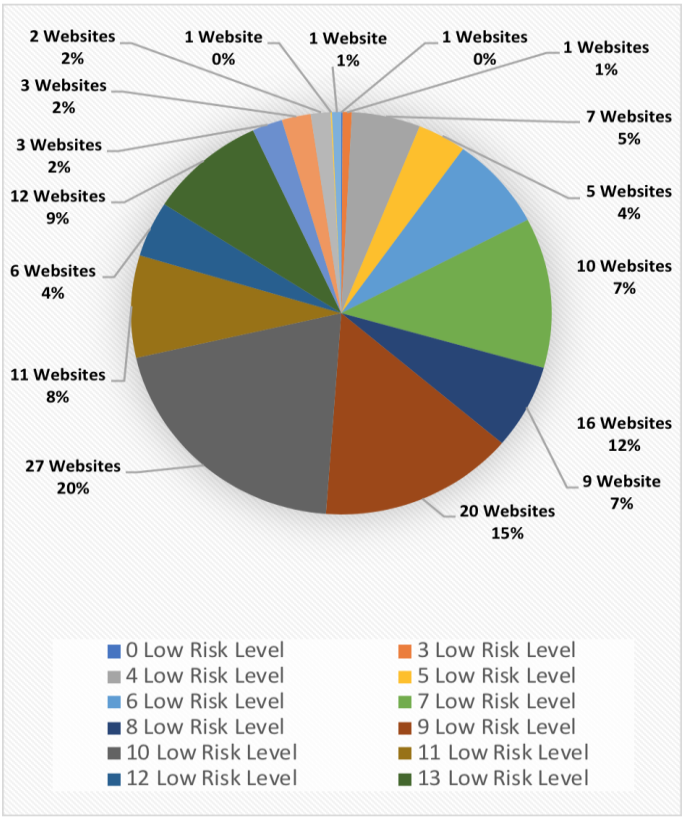

Fig. 3. Details of Low Risk Level for 135 tested websites

According to the diagram, most of the tested websites had Low Risk level vulnerabilities.

Details of Informational Risk level:

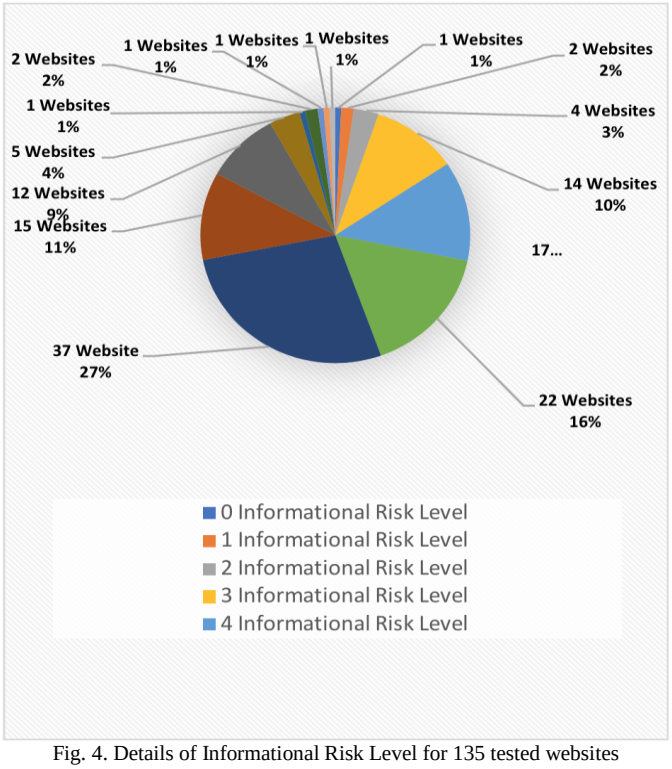

Fig. 4. Details of Informational Risk Level for 135 tested websites

According to the diagram, more than 60% of the tested websites had Informational-Risk Level vulnerabilities.

VI. Vulnerabilities Classification According to Risk Level

Classification of the vulnerabilities and risks provide a way to organize and communicate the results of the assessments and helps testers and project participants to prioritize issues, identify potential risks and tradeoff between the security issues exist in a web application.

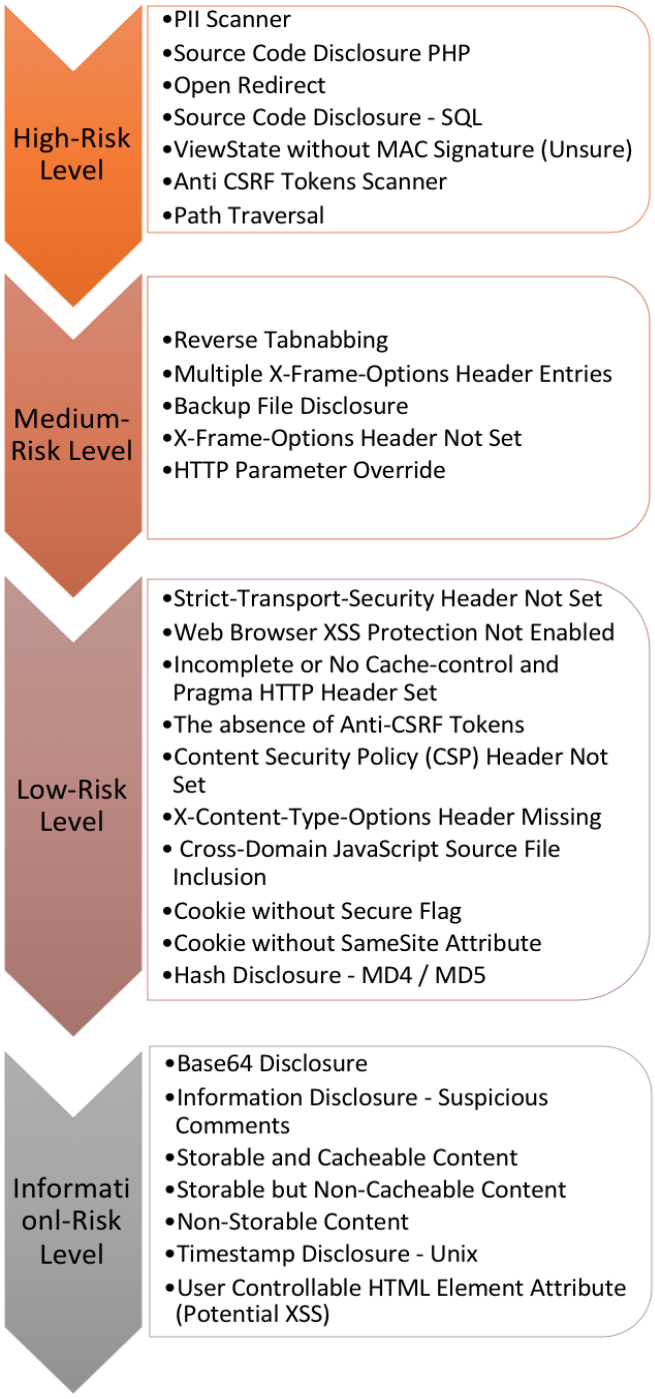

Fig. 5. Vulnerabilities Classification

VII. CONCLUSION

In this research the .AF web applications are selected as a target. Significant and valuable information is gathered which will be useful for hackers to exploit it. On the other hand, this information enable developer to develop, design and maintain the security of web applications by considering top vulnerabilities. The vulnerabilities are categorized as a high, medium, low and informational. The risk level of all categorizes are considerable even if it is informational.

In this study, 135 web applications are target against top vulnerabilities under the domain of Afghanistan. The results of this research with the ranking of the vulnerabilities shows, that security must be taking a serious attention.

REFERENCES

[1] OWASP Foundation, "OWASP Top Ten," 2020. [Online]. Available: https://owasp.org/www-project-top-ten/. [Accessed: 17-Apr-2020].
[2] A. Hamidi, M. M. Naier, and M. R. Bahez, "Evaluation of Web Application Security," Int. J. Eng. Appl. Sci. Technol.(ijeast), vol. 4, no. 11, pp. 5 (4 pp.)-5 (4 pp.), 2020.
[3] S. Panda and S. Ramani, "Protection of Web Application against Sql Injection Attacks," vol. 3, no. 1, pp. 166–168, 2013.
[4] V. Damjanovic and D. Djuric, "Functional Programming Way to Interact with Software Attacks and Vulnerabilities," in 2010 Third International Conference on Software Testing, Verification, and Validation Workshops, 2010, pp. 388–393.
[5] M. Vieira, N. Antunes, and H. Madeira, "Using Web Security Scanners to Detect Vulnerabilities in Web Services," in Proceedings of the International Conference on Dependable Systems and Networks, 2009, pp. 566–571.
[6] M. Vieira, N. Antunes, and H. Madeira, "Using web security scanners to detect vulnerabilities in web services," Proc. Int. Conf. Dependable Syst. Networks, no. August, pp. 566–571, 2009.
[7] B. S. Kennedy, "Common Web Application Vulnerabilities," 2005.
[8] O. Marques, "The world wide web," SpringerBriefs Comput. Sci., vol. 25, no. 9783319456973, pp. 3–18, 2016.
[9] J. Abirami, D. Ramalingam, and C. Valliyammai, "A top web security vulnerability SQL injection attack — Survey," 2015, pp. 1–9.
[10] O. Al-Khurafi and M. Al-Ahmad, "Survey of Web Application Vulnerability Attacks," 2015, pp. 154–158.
[11] B. Eshete, A. Villafiorita, and K. Weldemariam, "Early Detection of Security Misconfiguration Vulnerabilities in Web Applications," in 2011 Sixth International Conference on Availability, Reliability and Security, 2011, pp. 169–174.
[12] "OWASP API Security Project." [Online]. Available: https://owasp.org/www-project-api-security/. [Accessed: 18-Apr-2020].
[13] G. Wassermann and Z. Su, "Static detection of cross-site scripting vulnerabilities," in Proceedings - International Conference on Software Engineering, 2008, pp. 171–180.
[14] S. Kumar, R. Mahajan, N. Kumar, and S. K. Khatri, "A Study on Web Application Security and Detecting Security Vulnerabilities," 2017.
[15] V. Dehalwar, A. Kalam, M. L. Kolhe, and A. Zayegh, "Review of web-based information security threats in smart grid," 2017 7th Int. Conf. Power Syst. ICPS 2017, pp. 849–853, 2018.
[16] T. Lee, G. Won, S. Cho, N. Park, and D. Won, "Detection and Mitigation of Web Application Vulnerabilities Based on Security Testing," in Network and Parallel Computing, 2012, pp. 138–144.
[17] N. Shevchenko, T. A. Chick, P. O. Riordan, T. P. Scanlon, and C. Woody, "Threat Modeling : a Summary of Available Methods," Res. Rep., no. July, p. 26, 2018.
[18] S. Africa, M. Track, J. Jansen, and C. Centre, "Proceedings of the 13th International Conference on Cyber Warfare and Security, ICCWS 2018," Proc. 13th Int. Conf. Cyber Warf. Secur. ICCWS 2018, vol. 2018-March, no. March, p. 707, 2018.
[19] V. Sulkamo, "IoT from cyber security perspective Case study JYVSECTEC School of Technology, Communication and Transport Information Technology Master's Degree Programme in Cyber Security," no. June, 2018.
[20] C. J. Alberts and A. J. Dorofee, Managing information security risks : the OCTAVE approach. Addison-Wesley, 2003.